SET MATRICES AND THE PATH/CYCLE PROBLEM

SERGEY GUBIN

ABSTRACT. Presentation of set matrices and demonstration of their efficiency as a tool using the path/cycle problem.

INTRODUCTION

Set matrices are matrices whose elements are sets. The matrices comprise abilities of data storing and processing. That makes them a promising combinatorial structure. To prove the concept, this work applies the matrices to the path/cycle problem, see [1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, and many others]. The problem may be generalized as a problem to find all paths and all cycles of all length in form of vertex pairs (start, finish). That is a NP-hard problem because any of its solutions will include a solution of the Hamiltonian path/cycle problems [5]. This presentation uses set matrices to realize the following plan to solve the generalized problem: present the walk length dynamics with a generative grammar, but include in the grammar's production rules some path/cycle filters in order to deplete the resulting walk language to the indication of path/cycle's presence/absence, only.

The design's idea may be traced back trough the dynamic programming, the Ramsey theory, the formal language theory, and to the icosian calculus [16, 17]. Realization of the design requires to maintain a set of visited/unvisited vertices and to use that set as a filter in production of the next generation of walks. Set matrices satisfy the requirements. Sorting/factoring of the visited/unvisited vertices into vertex pairs (start, finish) creates a set matrix analog of the adjacency matrix. And the especially designed powers of the set matrix create an analytic path/cycle filter. The path/cycle language's specification gets a realization in form of the easy-to-check properties of the elements of the adjacency set matrix's powers.

The factoring of the set of visited/unvisited vertices into vertex pairs (start, finish) may be seen as a walk coloring where colors are the factor-sets. Then, the family of algorithms realizing the design can be parametrized with the following four extreme strategies: to color the walks with sets of the visited/unvisited start/finish vertices. Work [18] describes a walk coloring with the unvisited vertices. This work deploys walk coloring with the visited vertices.

Worst case for the algorithms is a complete graph. For a complete graph with n vertices, the algorithms perform n iterations and, on each of these iterations, $O(n^2)$-time processing for each of the n^2 vertex pairs. That totals in time $O(n^5)$ needed for the algorithms to find all paths and all cycles of all length in the form of vertex pairs (start, finish).

Date: September 17, 2007.
2000 *Mathematics Subject Classification.* Primary 05C38, Secondary 68R10.

1. Set Matrices

Let V be a universal set. Set matrices are matrices whose elements are sets. All set operations can be defined on the set matrices. For example, if $A = (a_{ij})$ and $B = (b_{ij})$ are set matrices of the appropriate sizes, then

Compliment:

$$A^c = (a_{ij}^c);$$

Join:

$$A \cup B = (a_{ij} \cup b_{ij});$$

Intersection:

$$A \cap B = (a_{ij} \cap b_{ij}),$$

Multiplication:

$$AB = (\bigcup_{\mu} a_{i\mu} \times b_{\mu j}),$$

- where "$\times$" is Cartesian product of sets, etc. More operations can be found in [18].

For the path/cycle problem, the most interesting operation is the set matrix multiplication. The operation can be redefined in different ways. In this presentation, let us use the following multiplication: for set matrices $A = (a_{ij})_{n\times m}$ and $B = (b_{ij})_{m\times k}$, product AB is the $n \times k$ set matrix whose elements are

$$(AB)_{ij} = \begin{cases} \bigcap_{\mu=1}^{n} a_{i\mu} \cup b_{\mu j}, & i \neq j \\ V, & i = j \end{cases} \tag{1.1}$$

Here and further, symbol $(X)_{ij}$ means (i,j)-element of matrix X.

Formula 1.1 is the formula of the number matrix multiplication, except "+" is replaced with "$\cap$", "$\times$" is replaced with "$\cup$", and some special cases are taken care of. The special cases treatment makes multiplication 1.1 a non-associative operation:

Exercise 1.1.

$$[\begin{pmatrix} \emptyset & \emptyset \\ \emptyset & \{a\} \end{pmatrix} \begin{pmatrix} \emptyset & \emptyset \\ \emptyset & \{b\} \end{pmatrix}] \begin{pmatrix} \emptyset \\ \{c\} \end{pmatrix} = \begin{pmatrix} V & \emptyset \\ \emptyset & V \end{pmatrix} \begin{pmatrix} \emptyset \\ \{c\} \end{pmatrix} = \begin{pmatrix} V \\ \emptyset \end{pmatrix},$$

$$\begin{pmatrix} \emptyset & \emptyset \\ \emptyset & \{a\} \end{pmatrix} [\begin{pmatrix} \emptyset & \emptyset \\ \emptyset & \{b\} \end{pmatrix} \begin{pmatrix} \emptyset \\ \{c\} \end{pmatrix}] = \begin{pmatrix} \emptyset & \emptyset \\ \emptyset & \{a\} \end{pmatrix} \begin{pmatrix} V \\ \emptyset \end{pmatrix} = \begin{pmatrix} V \\ \{a\} \end{pmatrix}.$$

Let A be a square set matrix. The following iterations define the left and right k-th powers of the matrix, $k \geq 1$:

$$R^1 = T^1 = A$$

$$(R^{k+1})_{ij} = \begin{cases} \bigcap_{\mu} (R^1)_{i\mu} \cup (R^k)_{\mu j}, & i \neq j \\ V, & i = j \end{cases} \tag{1.2}$$

$$(T^{k+1})_{ij} = \begin{cases} \bigcap_{\mu} (T^k)_{i\mu} \cup (T^1)_{\mu j}, & i \neq j \\ V, & i = j \end{cases}$$

Let us estimate the computational complexity of formula 1.2. Multiplication 1.1 requires $O(n^3)$ operations "$\cup$" and "$\cap$". Thus, if t_{k-1} is the number of operations needed to calculate $(k-1)$-th power, then the number of operations needed to calculate k-th power is

$$t_k = t_{k-1} + O(n^3) = O(kn^3).$$

Thus, the time needed to calculate k-th power can be estimated as

$$O(kn^3|V|). \tag{1.3}$$

The list of set matrix operations and properties can be continued. But let us start and demonstrate some benefits.

2. Path problem

Let $g = (V, A)$ be a given (multi) digraph: V is the vertex set and A is the arc set of g. Let the vertex set V be the universal set. Let's enumerate it:

$$V = \{v_1,\ v_2,\ \dots,\ v_n\}.$$

Let G be the adjacency matrix of g appropriate to this enumeration. Then the positive elements of powers of G indicate the presence of walks: vertex pairs (start, finish) of k-walks are indexes of positive elements of matrix G^k. The powers of this adjacency matrix can detect a shortest path but not a path of a specific length. Also, calculating the powers involves magnitudes of

$$O(n^{k-1}(\max_{ij}(G)_{ij})^k).$$

Although, the last problem can be solved with the Boolean adjacency matrices [18].

Let T be the following set matrix of size $n \times n$:

$$(T)_{ij} = \begin{cases} \{v_j\}, & (G)_{ij} > 0 \ \wedge\ i \neq j \\ V, & (G)_{ij} \leq 0 \ \vee\ i = j \end{cases} \tag{2.1}$$

Matrix T may be seen as an adjacency set matrix. Let T^k be the k-th right power of matrix T, defined with formulas 1.2.

Lemma 2.1. *In digraph g for $k < n$, if set $(T^k)_{ij} \neq V$, then the set is equal to*

$$(T^k)_{ij} = \bigcap_{\mu} \{v_{\mu_1}, v_{\mu_2}, \dots, v_{\mu_{k-1}}, v_{\mu_k}\},$$

where the intersection is taken over all ordered number samples

$$\mu = (\mu_1, \mu_2, \dots, \mu_{k-1}, \mu_k)$$

which satisfy the following constrains:

$$\begin{cases} 1 \leq \mu_x \leq n,\ x = 1, 2, \dots, k \\ (v_i, v_{\mu_1}) \in A,\ (v_{\mu_x}, v_{\mu_{x+1}}) \in A,\ x = 1, 2, \dots, k-1 \\ \mu_x \neq i,\ x = 1, 2, \dots, k \\ \mu_k = j \\ \mu_x \neq \mu_y \ \Leftrightarrow x \neq y \end{cases}$$

- where set A is the arc set of digraph g.

Proof. Due to definitions 1.2 and 2.1, if

$$(T^k)_{ij} = \bigcap_{\mu} (T^1)_{i\mu_1} \cup (T^1)_{\mu_1\mu_2} \cup \dots \cup (T^1)_{\mu_{k-2}\mu_{k-1}} \cup (T^1)_{\mu_{k-1}\mu_k} \neq V,$$

then there are number samples $\mu = (\mu_1, \mu_2, \dots, \mu_{k-1}, \mu_k)$ which satisfy the first four constrains, and

$$(T^k)_{ij} = \bigcap_{\mu} \{v_{\mu_1}\} \cup \{v_{\mu_2}\} \cup \dots \cup \{v_{\mu_{k-1}}\} \cup \{v_{\mu_k}\}, \tag{2.2}$$

where the intersection is taken over all those number samples. Proving the last constrain will prove the lemma. To do so, let's use mathematical induction over k.

For $k = 1$, due to definitions 1.2 and 2.1, $(T^1)_{ij} \neq V$ iff there are arcs from vertex v_i into vertex v_j and the arcs are not loops ($i \neq j$). Then, $(T^1)_{ij} = \{v_j\}$ and $(v_i, v_j) \in A$. Thus, the lemma holds for $k = 1$.

Because of an irregularity in the powers definition, the induction has to start from $k = 2$. In this case, due to definitions 1.2 and 2.1, if

$$(T^k)_{ij} = \bigcap_{\gamma} (T^1)_{i\gamma} \cup (T^1)_{\gamma j} \neq V,$$

then there are such indexes γ that

$$(T^k)_{ij} = \bigcap_{i\neq\gamma,\ \gamma\neq j,\ i\neq j,\ (v_i,v_\gamma)\in A,\ (v_\gamma,v_j)\in A} \{v_\gamma, v_j\},$$

where A is the arc set of digraph g. Thus, the lemma holds for $k = 2$.

Let's assume that the lemma holds for all $k \leq m-1 < n-1$, and let $(T^m)_{ij} \neq V$. Then, due to decomposition 2.2,

$$(T^m)_{ij} = \bigcap_{\mu} \{v_{\mu_1}\} \cup \{v_{\mu_2}\} \cup \ldots \cup \{v_{\mu_{m-1}}\} \cup \{v_{\mu_m}\} \neq V,$$

where the intersection is taken over some number samples μ, satisfying the first four constrains. Then, there is such number sample μ that

$$\{v_{\mu_1}\} \cup \{v_{\mu_2}\} \cup \ldots \cup \{v_{\mu_{m-1}}\} \cup \{v_{\mu_m}\} = Z \neq V.$$

Then, due to decomposition 2.2, for any of such number samples μ, the following holds:

$$(T^{m-1})_{i\mu_{m-1}} \subseteq \{v_{\mu_1}\} \cup \{v_{\mu_2}\} \cup \ldots \cup \{v_{\mu_{m-1}}\} \subseteq Z \neq V,$$

and

$$(T^{m-1})_{\mu_1\mu_m} \subseteq \{v_{\mu_2}\} \cup \{v_{\mu_3}\} \cup \ldots \cup \{v_{\mu_m}\} \subseteq Z \neq V.$$

Then, due to the induction hypothesis, both number samples

$$(\mu_1, \mu_2, \ldots, \mu_{m-2}, \mu_{m-1})$$

and

$$(\mu_2, \mu_3, \ldots, \mu_{m-1}, \mu_m)$$

satisfy all five constrains. Particularly,

$$\mu_x \neq \mu_y,\ \Leftrightarrow\ x \neq y,\ x, y = 1, 2, \ldots, m-1;$$

$$\mu_x \neq \mu_y,\ \Leftrightarrow\ x \neq y,\ x, y = 2, 3, \ldots, m;$$

and, due to the third constrain for $(T^{m-1})_{\mu_1\mu_m} \neq V$,

$$\mu_1 \neq \mu_m = j.$$

Thus, the whole number sample μ satisfies the fifth constrain. That concludes the induction and proves the lemma for all $k < n$. □

Lemma 2.1 allows the following interpretation:

Lemma 2.2. *In digraph g, if $(T^k)_{ij} \neq V$, then there is a k-path from vertex v_i into vertex v_j.*

Proof. The constrains in lemma 2.1 are the definition of a path from v_i into v_j. □

Lemmas 2.1 and 2.2 show that matrices T^k collect the vertex-bridges. That may be interesting for the graph toughness theory [6, 15].

Lemma 2.3. *In digraph g, if there is a k-path from vertex v_i into vertex v_j then $(T^k)_{ij} \neq V$.*

Proof. Let the following vertices constitute a k-path from vertex v_i into vertex v_j:

$$v_{\mu_1=i},\ v_{\mu_2},\ \ldots,\ v_{\mu_{k+1}=j}.$$

Indexes of these vertices satisfy the constrains in lemma 2.1. Then, due to definitions 1.2 and 2.1,

$$(T^k)_{ij} = \bigcap_{\mu} (T^1)_{i\mu_1} \cup (T^1)_{\mu_1\mu_2} \cup \ldots \cup (T^1)_{\mu_{k-2}\mu_{k-1}} \cup (T^1)_{\mu_{k-1}\mu_k} \subseteq$$

$$\subseteq \{v_{\mu_1}\} \cup \{v_{\mu_2}\} \cup \ldots \cup \{v_{\mu_k}\} \cup \{v_{\mu_{k+1}}\} \subseteq V - \{v_i\} \neq V.$$

□

Theorem 2.4. *In digraph g for $k \geq 1$, there are k-paths from vertex v_i into vertex v_j iff*

$$(T^k)_{ij} \neq V.$$

Proof. The theorem aggregates lemmas 2.2 and 2.3. Let us notice that case $k \geq n$ is covered by lemmas 2.1 and 2.3:

$$k \geq n \;\Rightarrow\; T^k = (V)_{n \times n}.$$

□

Estimation 1.3 shows the computational complexity to detect the k-paths with theorem 2.4. Particularly, when $k = n - 1$, the theorem detects the existence or absence of Hamiltonian paths in time

$$O(n^5).$$

All the results can be repeated with the left powers of matrix T. Also, definition 2.1 uses the arc finish vertices. Obviously, the results can be repeated with the start vertices using the following set matrix instead of matrix 2.1:

$$(R)_{ij} = \begin{cases} \{v_i\}, & (G)_{ij} > 0 \ \wedge\ i \neq j \\ V, & (G)_{ij} \leq 0 \ \vee\ i = j \end{cases} \tag{2.3}$$

Colorings 2.1 and 2.3 cover two of the four extreme strategies of walk coloring: to color walks with the visited start/finish vertices. Another two extreme strategies are discussed in [18]. They produce the same results but in terms of the compliment sets.

3. Cycle problem

Obviously, the solution of the path problem described in section 2 solves the cycle problem, as well. Let us formalize that analytically.

Let's define another set matrix multiplication: if A and B are set matrices of appropriate sizes, then

$$(AB)_{ij} = \begin{cases} \bigcap_{\nu} (A)_{i\nu} \cup (B)_{\nu j}, & i = j \\ V, & i \neq j \end{cases} \tag{3.1}$$

And let us define the following walk coloring:

$$(S)_{ij} = \begin{cases} \{\text{“Loop”}\}, & (G)_{ij} > 0 \ \wedge \ i = j \\ V, & (G)_{ij} \le 0 \ \vee \ i \ne j \end{cases},$$

$$S^1 = S, \ S^{k+1} = T^k R^1, \ k \ge 1, \tag{3.2}$$

- where set matrices T^k and R^1 were defined in section 2, and matrix multiplication 3.1 is used.

Theorem 3.1. *In digraph g for $k \ge 1$, there are k-cycles attached to vertex v_i iff*

$$(S^k)_{ii} \ne V.$$

Proof. Case when $k = 1$ is obvious. Let $k > 1$.

Necessity. Let a k-cycle be attached to vertex v_i, and let the cycle visit the following vertices in the order shown:

$$v_{\mu_1 = i}, \ v_{\mu_2}, \ \dots, \ v_{\mu_k}, \ v_{\mu_{k+1} = i}.$$

Then, the last k vertices in the row constitute a $(k-1)$-path from v_{μ_2} into v_i. Thus, due to lemma 2.1 and theorem 2.4,

$$v_i \in (T^{k-1})_{\mu_2 i} \ne V.$$

On the other hand, due to definition 2.3,

$$(R^1)_{i\mu_2} = \{v_i\} \ne V.$$

Thus, due to definition 3.1,

$$(S^k)_{ii} = ((T^{k-1})_{\mu_2 i} \cup \{v_i\}) \ \cap \dots \ \subseteq \ (T^{k-1})_{\mu_2 i} \cup \{v_i\} = (T^{k-1})_{\mu_2 i} \ \ne \ V.$$

Sufficiency. Let

$$(S^k)_{ii} = \bigcap_{\nu} (T^{k-1})_{i\nu} \cup (R^1)_{\nu i} \ne V.$$

Then, there is such number ν that

$$(T^{k-1})_{i\nu} \cup (R^1)_{\nu i} \ne V.$$

Then, due to theorem 2.4, there is a $(k-1)$-path from v_i into v_ν; and, due to definition 2.3, there is an arc from v_ν into v_i. The path and arc create a k-cycle attached to v_i. □

Estimation 1.3 gives the computational complexity of theorem 3.1. Particularly, when $k = n$, the theorem detects the existence/absence of Hamiltonian cycles in time $O(n^5)$. But some simplifications are possible. The existence/absence of Hamiltonian cycles can be detected by only calculating any one string of matrix T^{n-1}. That reduces the time needed to solve the Hamiltonian cycle problem to

$$O(n^4).$$

Conclusion

The paper presented set matrices as an efficient tool for solving the combinatorial problems. The matrices were used to solve the path/cycle problem in polynomial time:

k-path: Calculate set matrix T^k with formulas 2.1 and 1.2. Use theorem 2.4 to detect all k-paths in form vertex pair (start, finish);

k-cycle: Calculate set matrix S^k with formulas 2.1, 1.2, 2.3, 3.1, and 3.2. Use theorem 3.1 to detect all vertices which have a k-cycle attached.

Boolean property "It is equal to the vertex set" of the elements of matrices T^k and S^k fulfill the path/cycle language's specification: indicate the presence/absence of paths/cycles. For a graph with n vertices, it will take $O(n^5)$-time to write down the whole language in form of $O(n)$ matrices of size $n \times n$ filled with 1 and 0: 1 will mean the existence of appropriate paths/cycles and 0 will mean their absence.

References

[1] W.T. Tutte. On Hamiltonian Circuits. J. London Math. Soc. 21, 98-101, 1946.
[2] W.T. Tutte. A Theorem on Planar Graphs. Trans. Amer. Math. Soc. 82(1956), 99-119.
[3] O.A. Ore. A note on Hamiltonian Circuits. Amer. Math. Monthly. 67(1960), 55.
[4] Stephen Cook. The complexity of theorem-proving procedures. In Conference Record of Third Annual ACM Symposium on Theory of Computing, 151-158, 1971
[5] Richard M. Karp. Reducibility among combinatorial problems. In R.E. Miller and J.W. Thatcher, editors, Complexity of Computer Computations, 85-103, New York, 1972 Plenum Press.
[6] Chvatal, Vaclav. Tough graphs and Hamiltonian circuits. Discrete Mathematics 5 (3): 215228, 1973
[7] H.A. Jung. On Maximal Circuits in Finite Graphs. Annals of Discrete Math 3(1978), 129-144.
[8] B. Bollabas and A.M. Hobbs. Hamiltonian cycles. Advances in Graph Theory. (B. Bollabas ed) North-Holland Publ., Amsterdam, 1978, 43-48.
[9] M.R. Garey and D.S. Johnson. Computers and Intractability, a Guide to the Theory of NP-Completeness. W.H. Freeman and Co,. San Francisco, 1979.
[10] G.H. Fan. A new Sufficient Condition for cycles in graphs. J. Combinat. Theory B37(1984) 221-227.
[11] Mihalis Yannakakis. Expressing combinatorial optimization problems by linear programs. In Proc. of the twentieth annual ACM Sympos. on Theory of computing. Chicago, Illinois, 223 - 228, 1988.
[12] D. Bauer, E. Schmeichel, and H.J. Veldman. Some Recent Results on Long Cycles in Tough Graphs. Off Prints from Graph Theory, Combinatorics, and Applications. Ed. Y. Alavi, G. Chartrand, O.R. Ollerman, A.J. Schwenk. John Wiley and Sons, Inc. 1991.
[13] R. Diestel. Graph Theory. New York, Springer, 1997.
[14] The Traveling Salesman Problem and Its Variations. Gregory Gutin and Abraham P. Punnen (Eds.). Kluwer Academic Publishers, 2002.
[15] Bauer, Douglas; Broersma, Hajo; Schmeichel, Edward. Toughness in graphs a survey. Graphs and Combinatorics 22 (1): 135, 2006.
[16] Hamilton, William Rowan. Memorandum respecting a new system of roots of unity. Philosophical Magazine, 12 1856
[17] Hamilton, William Rowan. Account of the Icosian Calculus. Proceedings of the Royal Irish Academy, 6 1858
[18] Sergey Gubin. Finding paths and cycles in graphs. E-print arXiv:0709.0974, arXiv.org, 2007

Genesys Telecommunication Laboratories, Inc.
Current address: 1255 Treat Blvd., Walnut Creek, CA 94596
E-mail address: sgubin@genesyslab.com